Title: Security Requirements of Diagnostic Communication

Confidentiality level: Alliance Internal

Signature (Alliance Single Technical Leader)

Signature (Renault validator)

Signature (Nissan validator)

Revisions

Matsuyama	Yasunaga	XX6	2021-9-22	Change automatic lock from 80km to 500km	1.1	A	N
Miyashita	Yasunaga	XW3	2020-10-06	Newly established	1.0	A	N
APPROVED BY	AUTHOR	SECT	APPROVED DATE	ALTERATION	CHG RNDS	Ver GDNo rmes	CHG NDS

RENAULT-NISSAN

Important Notices and Disclaimers

- This document is based on NISSAN document “Security Requirements for ECUs using diagnostic communication”.
  It complies with the agreement reached between RENAULT and NISSAN in 2020-10-06.
  Any revision or alteration of this document is subject to prior approval by the RENAULT or NISSAN secretariat.
- The original version of this document was written in English.
  In the event of any discrepancies or differences in meaning created as a result of translation of this document into a foreign language, the meaning found in the English original shall take precedence.

Foreword

Renault	Issued by	:	Ivan Studnia
	Validated by	:	Pascal Chevalier
NISSAN	Issued by	:	Hiroki Yasunaga (XX6)
	Validated by	:	Satoru Matsuyama (XX6),

Contents

Introduction

This document specifies the security requirements of each diagnostic requirement defined in the [REF1] with an attempt to call attention.

Security Requirements for ECUs using diagnostic communication

1. Scope

This document is the minimum security requirements for all the ECUs using diagnostic communication (UDS) with the diagnostic tool. When the client is an ECU, the security requirements defined in this specification can also be applied by the ECUs which act as the server. If there were more secure and reasonable mitigations or requirements for the ECU(s), suppliers or ECU designers could apply them.

Mainly the security requirements defined in this document is for the post-production phase. If the security requirements caused any trouble during the pre-production or production phases, these security requirements could be disabled temporarily (refer to Section 11). In detail, this document request that safety related diagnostic communication shall be protected by SecurityAccess (Asynmetric), vehicle safety condition … etc. SecurityAccess (Synmetric) is not recommended. And this document request digital signature for firmeware. This document shall not want to block other regulation.

2. Normative and informative references

These normative and informative references apply the latest version.

Normative

— [REF1] ISO14229-1: 2020 Road vehicles - Unified diagnostic services (UDS) - Part 1: Application layer
— [REF2] NIST Special Publication 800-57 : Recommendation for Key Management
— [REF3] PKCS#1 v1.5 RSA Cryptography Standard
— [REF4] PKCS#1 v2.2 RSA Cryptography Standard
— [REF5] Advanced Encryption Standard (AES) (FIPS PUB 197)
— [REF6] NIST Special Publication 800-38A
— [REF7] ISO11898: Road vehicles – Controller Area Network (CAN)
— [REF8] ISO26262: Road vehicles – Functional safety
— [REF9] RNDS-C-00065 v1.0 - Unified Diagnostic Services (UDS) Implementation
— [REF10] RNDS-C-00069_2.0 - Standard Security Access
— [REF11] KD2-43917 [8] - Vehicle level reliability assurance standard
— [REF12] [Step 2 FOTA] FOTA Communication Sequence Specification
— [REF13] ISO13400: 2012 Road vehicles — Diagnostic communication over Internet Protocol layer
— [REF14] AUTOSAR Specification of Synchronized Time-Base Manager
— [REF15] Digital Signature Standard (DSS) (FIPS PUB 186-4)
— [REF16] Generic Security Requirements of SW Update for Target-ECUs using diagnostic and FOTA communication
— [REF17] RNDS-C-00073_4.0 – Rules for the management of on-board Data Identifiers in UDS implementation

3. Terms and definitions

For the purpose of this standard, the following terms and definitions apply.

AES

Standard Advanced Encryption Standard cryptographic algorithm as defined in the FIPS 197 (Refer to [REF5]).

Availability

The data can be accessed and available when an authorized entity requests it.

Brute force attack

In cryptography, a brute-force attack consists of an attacker submitting many passwords or passphrases with the hope of eventually guessing correctly. The attacker systematically checks all possible passwords and passphrases until the correct one is found. Alternatively, the attacker can attempt to guess the key which is typically created from the password using a key derivation function. This is known as an exhaustive key search.

CBC mode

Cipher Block Chaining mode, as specified in [REF6] is that the plaintext input must consist of a sequence of blocks.

Confidentiality

Data is not available or disclosed to unauthorized individuals, entities, or processes

Client

Client is the function that makes use of the diagnostic services.

Diagnostic service

Information exchange is initiated by a diagnostic tool in order to require diagnostic information for an ECU or/and to modify its behavior for diagnostic purpose.

Diagnostic session

Diagnostic session states within the ECU in which a specific set of diagnostic services and functionality is enabled.

Diagnostic tool

The diagnostic tool is used to control functions such as test, inspection, monitoring or diagnosis of an on-vehicle electronic control unit.

High confidentiality

The disclosure of the high confidentiality data will lead to severe confidential issues, like firmware, private keys etc..

High privacy

The disclosure of the high privacy data will lead to severe privacy issues, like personally identifiable information and sensitive data.

Integrity

The accuracy and completeness of data is maintained and assured over its entire lifecycle.

Dictionary attack

A dictionary attack is based on trying all the strings in a pre-arranged listing, typically derived from a list of words such as in a dictionary.

Predictive analytics cyber-attack

A predictive analytics cyber-attack is using a variety of statistical techniques from data mining, predictive modelling, and machine learning that analyze current and historical facts to make predictions about future or otherwise unknown events.

Replay attack

A replay attack is that a valid data transmission is maliciously or fraudulently repeated or delayed.

Safe condition

The safe condition is the state will not cause grade ▽A safety issues. Refer to [REF11] for the detail of failure grade ▽A.

The safe conditions must be specified by suppliers or ECU designers.

Safety ECUs

ECUs with ASIL are safety ECUs.

Exploitation of safety ECUs will infringe the safety goal and have a negative impact on safety.

Server

Server is the function that is part of an ECU that provides the diagnostic services.

Threat

A potential cause of an unwanted incident, which may result in harm to a system.

Authentication (0x29) service

The Authentication service is used by the diagnostic tool to provide a means for the client to prove its identity, allowing it to access data and/or diagnostic services, which have restricted access for, for example security, emissions, or safety reasons.

SecuredDataTransmission (0x84) service

The SecuredDataTransmission service is used by the diagnostic tool, if a client intends to use diagnostic services defined in a secured mode.

AccessTimingParameter (0x83) service

The AccessTimingParameter service is used by the diagnostic tool to read/ modify the timing parameters for an active communication.

ClearDiagnosticInformation (0x14) service

The ClearDiagnosticInformation service is used by the diagnostic tool to clear diagnostic information in one or multiple ECUs' memory.

CommunicationControl (0x28) service

The CommunicationControl service is used to switch on /off the transmission and /or the reception of certain messages of (an) ECU(s).

ControlDTCSetting (0x85) service

The ControlDTCSetting service is used by the diagnostic tool to stop or resume the updating of DTC status bit in the ECU(s).

DiagnosticSessionControl (0x10) service

The DiagnosticSessionControl service is used to enable different diagnostic sessions in the ECU(s)

DynamicallyDefineDataIdentifier (0x2C) service

The DynamicallyDefineDataIdentifier service allows the diagnostic tool to dynamically define in an ECU a data memory data identifier that can be read via the ReadDataByIdentifier service at a later time.

ECUReset (0x11) service

The ECRReset service is used by the diagnostic tool to request an ECU reset.

InputOutputControlByIdentifier (0x2F) service

The InputOutputControlByIdentifier service is used by the diagnostic tool to substitute a value for an input signal, internal ECU function and/or force control to a value for an output (actuator) of an electronic system.

LinkControl (0x87) service

The LinkControl service is used to control the communication between the diagnostic tool and the ECU(s) in order to gain bus bandwidth for diagnostic purpose.

ReadDataByIdentifier (0x22) service

The ReadDataByIdentifier service allows the diagnostic tool to request data record values from the ECU identified by one or more dataIdentifiers.

ReadDataByPeriodicIdentifier (0x2A) service

The ReadDataByPeriodicIdentifier service allows the diagnostic tool to request the periodic transmission of data record values from the ECU identified by one or more periodicDataIdentifiers.

ReadDTCInformation (0x19) service

The ReadDTCInformation service allows the diagnostic tool to read the status of ECU resident Diagnostic Trouble Code (DTC) information from any ECU or ECUs within a vehicle.

ReadMemoryByAddress (0x23) service

The ReadMemoryByAddress service allows the diagnostic tool to request data memory data from the ECU via provided starting address and size of memory to be read.

ReadScalingDataByIdentifier (0x24) service

The ReadScalingDataByIdentifier service allows the diagnostic tool to request scaling data record information for the ECU identified by a dataIdentifier.

ResponseOnEvent (0x86) service

The ResponseOnEvent service requests an ECU to start or stop transmission of responses on a special event.

RequestDownload (0x34) service

The RequestDownload service is used by the diagnostic tool to request the negotiation of a data transfer from the diagnostic tool to the ECU (download).

RequestFileTransfer (0x38) service

The RequestFileTransfer service is used by the diagnostic tool to initiate a file data transfer from either the diagnostic tool to the ECU or from the ECU to the diagnostic tool (download or upload). Additionally, this service has capabilities to retrieve information about the file system.

RequestTransferExit (0x37) service

The RequestTransferExit service is used by the diagnostic tool to request the termination of a data transfer between the diagnostic tool and ECU (upload or download).

RequestUpload (0x35) service

The RequestUpload (0x35) service is used by the diagnostic tool to request the negotiation of a data transfer from the ECU to the diagnostic tool (upload).

RoutineControl (0x31) service

The RoutineControl service is used by the diagnostic tool to execute a defined sequence of steps and obtain any relevant results.

SecuredDataTransmission (0x84) service

The SecuredDataTransmission service is applicable if a diagnostic tool intends to use diagnostic services defined in this document in a secured mode.

SecurityAccess (0x27) service

The purpose of this service is to provide a means to access data and/or diagnostic services, which have restricted access for security, emissions, or safety reasons. Refer to [REF10].

TransferData (0x36) service

The TransferData service is used by the diagnostic tool to transfer data either from the diagnostic tool to the

ECU (download) or requests data from the ECU to the diagnostic tool (upload).

TesterPresent (0x3E) service

The TesterPresent service is used to indicate an ECU (or ECUs) that the diagnostic tool is still connected to the vehicle and that certain diagnostic services and /or communications that have been previously activated are to remain active.

WriteDataByIdentifier (0x2E) service

The WriteDataByIdentifier service allows the diagnostic tool to write information into the ECU at an internal location specified by the provided data identifier.

WriteMemoryByAddress (0x3D) service

The WriteMemoryByAddress service allows the diagnostic tool to write information into the ECU at one or more contiguous memory locations

4. Symbols and abbreviations

For the purpose of this standard, the following symbols and abbreviated terms apply.

AT

Automatic Transmission

BCM

Body Control Module

CAN

Controller Area Network

DoCAN

Diagnostic communication over Controller Area Network

DoIP

Diagnostics over Internet Protocol

DTC

Diagnostic Trouble Code

ECM

Engine Control Module

ECU

Electronic Control Unit

FOTA

Firmware Over-The-Air

GW

Security Gateway

IVI

In Vehicle Infotainment

IVC

In Vehicle Connectivity

OBD

On-board diagnostics

PT

Production trial

VC

Vehicle confirmation

5. Assumed security architectures and cyber-attacks

This section describes assumed security architectures and cyber-attacks.

5.1 Assumed security architectures

This section describes the assumed security architectures. Fig.1 shows the architecture with GW ECU, which is a more secure architecture using defense in depth. In this target security architecture, the GW ECU provides filtering and authentication capabilities. It represents a first layer of defense to protect safety-critical functions against a malicious Diagnostic tool or OBD Dongle, as well as attacks from exposed ECUs like infotainment and connectivity. Nonetheless, in case an attacker may compromise the GW ECU, there is not an appropriate protection on target ECUs (e.g. ECM, AT, BCM) against malicious UDS frames forged by the attacker. Protection of safety-critical ECUs is thus solely relying on the protection of the GW ECU, which is not compatible with the defense in-depth security policy applied by the Alliance.

The target of this specification is to provide a minimal set of security requirements to be implemented by ECUs connected behind the GW ECU, and which aim to protect privileged UDS services through strong authentication and secure firmware update.

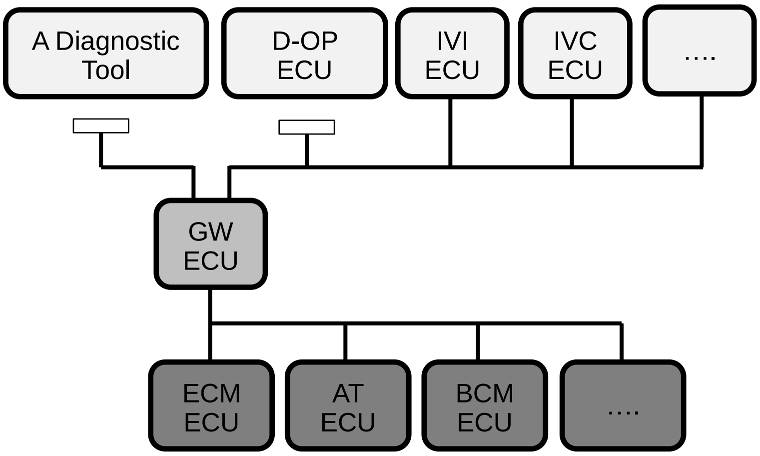
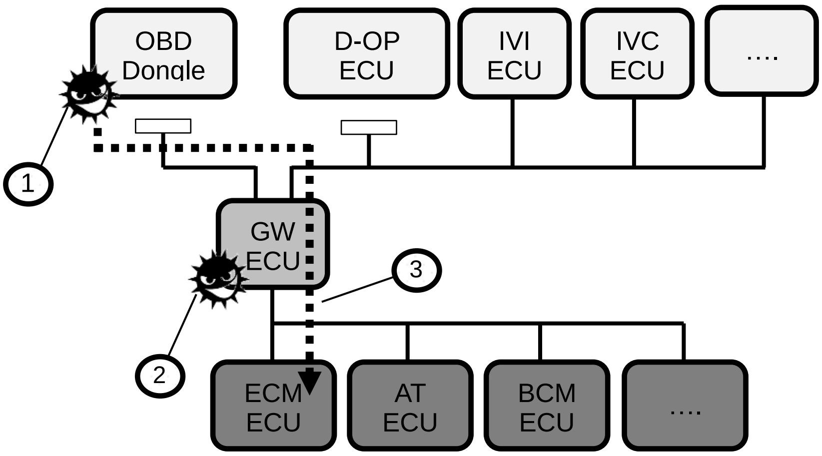


Fig.1 — The architecture with GW ECU

5.2 Assumed cyber-attacks

This section describes some examples of assumed security attacks. The attack shown in Fig.2, 3 is attack during vehicle running on road.

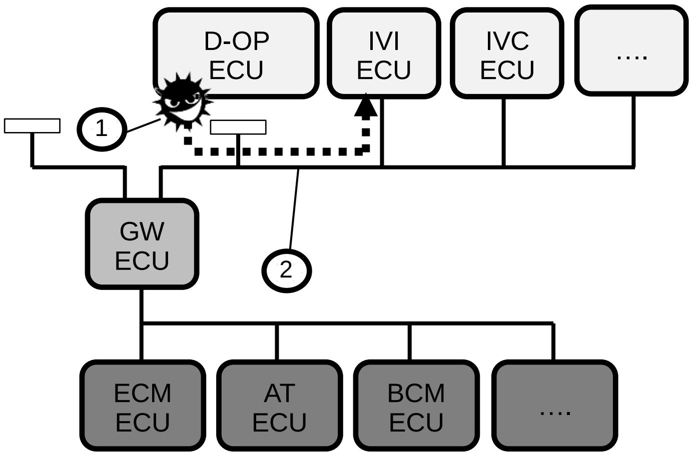


1 OBD Dongle is compromised.

2 GW ECU is compromised and either access control of GW ECU is disabled, or GW is used to forge malicious UDS requests

3 OBD Dongle sends high risk diagnostic services when the vehicle is in high risk condition.

Fig.2 — Attacks via OBD dongle during vehicle running



1 D-OP ECU is compromised.

2 D-OP ECU sends high risk diagnostic services when the vehicle is in high risk condition.

Fig.3 — Attacks via D-OP ECU during vehicle running

6. Security requirements for diagnostic services

This section describes the security requirements for each diagnostic service defined in the [REF1 and REF9] against the threat which might lead to unsafety.

Refer to [REF12] for the details of diagnostic services allowed in FOTA session.

6.1 Requirement terminology

For each diagnostic service, the functional requirement and technical requirement are defined with the unique identifier as the following table.

The threat with unique threat identifier are described to make the requirements be understandable.

Flexibility is defined for each technical requirement based on different kinds of ECUs.

<table>
<tr><td colspan="13">Threat</td></tr>
<tr><td>TH_C_ID(X)</td><td colspan="12">TH_Text</td></tr>
<tr><td>TH_I_ID(X)</td><td colspan="12">TH_Text</td></tr>
<tr><td>TH_A_ID(X)</td><td colspan="12">TH_Text</td></tr>
<tr><td>TH_ID(X)</td><td colspan="12">TH_Text</td></tr>
<tr><td colspan="4">Security requirements</td><td colspan="9">Flexibility</td></tr>
<tr><td rowspan="4">FUNC_ID</td><td rowspan="4">Functional Requirement</td><td rowspan="4">TECH_ID</td><td rowspan="4">Technical Requirement</td><td colspan="6">F_ECU</td><td colspan="3">NF_ECU</td></tr>
<tr><td colspan="4">S_ECU</td><td colspan="2">NS_ECU</td><td colspan="2">S_ECU</td><td rowspan="3">NS_ECU</td></tr>
<tr><td colspan="2">F_Se</td><td colspan="2">DT_Se</td><td rowspan="2">F_Se</td><td rowspan="2">DT_Se</td><td rowspan="2">SR</td><td rowspan="2">NS</td></tr>
<tr><td>SR</td><td>NS</td><td>SR</td><td>NS</td></tr>
<tr><td>FUNC_ID(X)</td><td>FUNC_Text</td><td>TECH_ID(X)</td><td>TECH_Text</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>

TH_ID(X): is the unique identifier of the potential threat which may be against confidentiality, integrity and availability. “(X)” is history identification for update.

TH_C_ID(X): is the unique identifier of the threat against confidentiality. “(X)” is history identification for update.

TH_I_ID(X): is the unique identifier of the threat against integrity. “(X)” is history identification for update.

TH_A_ID(X): is the unique identifier of the threat against availability. “(X)” is history identification for update.

TH_Text: is a description of the threat.

FUNC_ID(X): is the unique identifier of the functional requirement. “(X)” is history identification for update.

TECH_ID(X): is the unique identifier of the technical requirement. “(X)” is history identification for update.

FUNC_Text: is a description of the functional requirement.

TECH_Text: is a description of the technical requirement.

F_ECU: are FOTA-ECUs. Refer to [REF12] for details.

NF_ECU: are other ECUs than FOTA-ECUs.

S_ECU: are safety ECUs. Refer to the definition of safety ECU in the Section 3 for details.

NS_ECU: are other ECUs than safety related ECUs.

F_Se: is FOTA session enabled by FOTA functions.

DT_Se: are the diagnostic sessions except FOTA session and systemSupplierSpecific session which are enabled by the diagnostic tool.

SR: are safety related data, functions, routines and etc..

NS: are others than safety related data, functions, routines and etc..

Flexibility:

CF0: one of the defined requirements has to be met (must)

F0: no flexibility, requirement has to be met (must)

F1: low flexibility, requirement barely negotiable (want)

NA: No applicable request

6.2 DiagnosticSessionControl (0x10) service

Threat					
TH_C_1000(1)	Out of scope				
TH_I_1000(1)	There is no threat for unsafety.				
Security requirements				Flexibility	
FUNC_ID	Functional Requirement	TECH_ID	Technical Requirement	S_ECU	NS_ECU
No Requirement					

Threat					
TH_A_1001(1)	An attacker can launch ProgrammingSession to disable ECU functions				
Security requirements				Flexibility	
FUNC_ID	Functional Requirement	TECH_ID	Technical Requirement	S_ECU	NS_ECU
FUNC_101x(1)	Control the service access	TECH_1011(1)	The ECU SHALL be able to start ProgrammingSession after getting unlock status by SecurityAccess 0x27 (Refer to chapter 10)	F0	F0
Note1: If TECH_1011 is applied and this service is used in plant, ECU designers shall apply Virgin mode (Refer to Section 11).					

6.3 ECUReset (0x11) service

Threat					
TH_C_1100(1)	Out of scope				
TH_I_1100(1)	There is no threat for unsafety.				
Security requirements				Flexibility	
FUNC_ID	Functional Requirement	TECH_ID	Technical Requirement	S_ECU	NS_ECU
No Requirement					

Threat					
TH_A_1101(1)	An attacker can use ECU reset service when the vehicle is running. When the ECU program is being rebooted during the ECU reset, most of ECUs lose the control of their normal functions, which might lead to unsafety.				
Security requirements				Flexibility	
FUNC_ID	Functional Requirement	TECH_ID	Technical Requirement	S_ECU	NS_ECU
FUNC_110x(1)	Determine the execution conditions	TECH_1101(1)	In Default Session, Extended Session, the ECU SHALL be able to start ECUReset service, only when the vehicle is in the safe condition.	CF0	NA
FUNC_112x(2)	Control the service access and determine the execution conditions	TECH_1121(2)	In Extended Session, the ECU SHALL be able to start ECUReset service after getting unlock status by SecurityAccess 0x27 (Refer to chapter 10). In Default session, ECU shall not be able to start ECUreset without the safe condition.	CF0	NA
FUNC_111x(1)	Determine the execution conditions	TECH_1111(1)	In FOTA Session, the ECU SHALL follow FOTA specification.	F0	F0
Note1: (TECH_1101 or TECH_1121) and TECH_1111 has/have to be met. Note2: TECH_1101 and TECH_1111 is recommended if this service/parameter is used in no network environment. Note3: If TECH_1121 is applied and this service is used in plant, ECU designers shall apply Virgin mode (Refer to Section 11).					

6.4 SecurityAccess (0x27) service

Threat					
TH_C_2700(1)	Out of scope				
TH_I_2700(1)	There is no threat for unsafety.				
TH_A_2700(1)	There is no threat for unsafety.				
Security requirements				Flexibility	
FUNC_ID	Functional Requirement	TECH_ID	Technical Requirement	S_ECU	NS_ECU
FUNC_270x(1)	Apply kind of SecurityAccess	TECH_2700(2)	ECU shall implement RSA 2048 base (refer to 10.3) ~~or ECDSA 256 base (refer to 10.4). (Both cases shall be estimated to understand development estimation.)~~	F0	F0

6.5 TesterPresent (0x3E) service

Threat					
TH_C_3E00(1)	Out of scope				
TH_I_3E00(1)	There is no threat for unsafety.				
TH_A_3E00(1)	There is no threat for unsafety.				
Security requirements				Flexibility	
FUNC_ID	Functional Requirement	TECH_ID	Technical Requirement	S_ECU	NS_ECU
No Requirement					

6.6 ReadDataByIdentifier (0x22) service

Threat					
TH_C_2200(1)	Out of scope				
TH_I_2200(1)	There is no threat for unsafety.				
TH_A_2200(1)	There is no threat for unsafety.				
Security requirements				Flexibility	
FUNC_ID	Functional Requirement	TECH_ID	Technical Requirement	S_ECU	NS_ECU
No Requirement					

6.7 ReadMemoryByAddress (0x23) service

Threat					
TH_C_2301(1)	The service might read confidential data, which might lead to compromise SecurityAccess .				
Security requirements				Flexibility	
FUNC_ID	Functional Requirement	TECH_ID	Technical Requirement	S_ECU	NS_ECU
FUNC_230x(1)	forbidden to use the service	TECH_2301(1)	The ECU SHALL not be able to use ReadMemoryByAddress service. NRC is responded.	F0	F0

Threat					
TH_I_2300(1)	There is no threat for unsafety.				
TH_A_2300(1)	There is no threat for unsafety.				
Security requirements				Flexibility	
FUNC_ID	Functional Requirement	TECH_ID	Technical Requirement	S_ECU	NS_ECU
No Requirement					

6.8 DynamicallyDefineDataIdentifier (0x2C) service

Threat					
TH_C_2C00(1)	Out of scope				
TH_A_2C00(1)	There is no threat for unsafety.				
Security requirements				Flexibility	
FUNC_ID	Functional Requirement	TECH_ID	Technical Requirement	S_ECU	NS_ECU
No Requirement					

Threat					
TH_I_2C01(1)	The settable parameters might be set illegitimately, which might lead to unsafety.				
Security requirements				Flexibility	
FUNC_ID	Functional Requirement	TECH_ID	Technical Requirement	S_ECU	NS_ECU
FUNC_2C0x(1)	forbidden to use the service	TECH_2C01(1)	The ECU SHALL not be able to use DynamicallyDefineDataIdentifier service. NRC is responded.	F0	F0

6.9 WriteDataByIdentifier (0x2E) service

Threat					
TH_C_2E00(1)	Out of scope				
TH_A_2E00(1)	There is no threat for unsafety.				
Security requirements				Flexibility	
FUNC_ID	Functional Requirement	TECH_ID	Technical Requirement	S_ECU	NS_ECU
No Requirement					

Threat									
TH_I_2E01(1)	The settable parameters might be set illegitimately, which might lead to unsafety.								
Security requirements				Flexibility					
				S_ECU				NS_ECU	
				F_Se		DT_Se			
FUNC_ID	Functional Requirement	TECH_ID	Technical Requirement	SR	NS	SR	NS	F_Se	DT_Se
FUNC_2E0x(1)	Control the service access	TECH_2E01(1)	The ECU SHALL be able to start WriteDataByIdentifier service after getting unlock status by SecurityAccess 0x27 (Refer to chapter 10)	NA		F0		NA	F0
Note1: TECH_2E01 is not applicable during FOTA session ~~Note2: TECH_2E01 is not applicable for support check.~~ Note3: If TECH_2E01 is applied and this service is used in plant, ECU designers shall apply Virgin mode (Refer to Section 11).									

6.10 WriteMemoryByAddress (0x3D) service

Threat					
TH_C_3D00(1)	Out of scope				
TH_A_3D00(1)	There is no threat for unsafety.				
Security requirements				Flexibility	
FUNC_ID	Functional Requirement	TECH_ID	Technical Requirement	S_ECU	NS_ECU
No Requirement					

Threat					
TH_I_3D01(1)	The settable parameters might be set illegitimately, which might lead to unsafety.				
Security requirements				Flexibility	
FUNC_ID	Functional Requirement	TECH_ID	Technical Requirement	S_ECU	NS_ECU
FUNC_3D0x(1)	forbidden to use the service	TECH_3D01(1)	The ECU SHALL not be able to use WriteMemoryByAddress service. NRC is responded.	F0	F0

6.11 ClearDiagnosticInformation (0x14) service

Threat					
TH_C_1400(1)	Out of scope				
TH_I_1400(1)	There is no threat for unsafety.				
TH_A_1400(1)	There is no threat for unsafety.				
Security requirements				Flexibility	
FUNC_ID	Functional Requirement	TECH_ID	Technical Requirement	S_ECU	NS_ECU
No Requirement					

6.12 ReadDTCInformation (0x19) service

Threat					
TH_C_1900(1)	Out of scope				
TH_I_1900(1)	There is no threat for unsafety.				
TH_A_1900(1)	There is no threat for unsafety.				
Security requirements				Flexibility	
FUNC_ID	Functional Requirement	TECH_ID	Technical Requirement	S_ECU	NS_ECU
No Requirement					

6.13 InputOutputControlByIdentifier (0x2F) service

Threat					
TH_C_2F00(1)	Out of scope				
TH_I_2F00(1)	There is no threat for unsafety.				
Security requirements				Flexibility	
FUNC_ID	Functional Requirement	TECH_ID	Technical Requirement	S_ECU	NS_ECU
No Requirement					

Threat						
TH_A_2F01(1)	When the vehicle is moving, the illegitimate control of actuators might lead to unexpected action of actuators, which might lead to unsafety.					
Security requirements				Flexibility		
				S_ECU		NS_ECU
FUNC_ID	Functional Requirement	TECH_ID	Technical Requirement	SR	NS	
FUNC_2F1x(1)	Determine the execution conditions	TECH_2F11(1)	The ECU SHALL be able to start InputOutputControlByIdentifier service, only when the vehicle is in the safe condition.	CF0	NA	NA
		TECH_2F12(1)	The ECU SHALL be able to stop InputOutputControlByIdentifier service at the timing when the state of the vehicle is changed to any other conditions than the safe condition.			
FUNC_2F2x(1)	Control the service access	TECH_2F21(1)	The ECU SHALL be able to start InputOutputControlByIdentifier service after getting unlock status by SecurityAccess 0x27 (Refer to chapter 10)	CF0	NA	NA
Note1: (TECH_2F11 and TECH_2F12) or TECH_2F21 has/have to be met. Note2: (TECH_2F11 and TECH_2F12) is recommended if this service/parameter is used in no network environment. ~~Note3: TECH_2F21 is not applicable for support check.~~ Note4: If TECH_2F21 is applied and this service is used in plant, ECU designers shall apply Virgin mode (Refer to Section 11).						

6.14 RoutineControl (0x31) service

Threat					
TH_C_3100(1)	Out of scope				
Security requirements				Flexibility	
FUNC_ID	Functional Requirement	TECH_ID	Technical Requirement	S_ECU	NS_ECU
No Requirement					

Threat						
TH_I_3101(1)	The settable parameters might be set illegitimately when the routine is started, which might lead to unsafety.					
Security requirements				Flexibility		
				S_ECU		
FUNC_ID	Functional Requirement	TECH_ID	Technical Requirement	SR	NS	NS_ECU
FUNC_311x(1)	Control the service access	TECH_3111(1)	The ECU SHALL be able to start RoutineControl service after getting unlock status by SecurityAccess 0x27 (Refer to chapter 10)	F0	NA	NA
Note1 TECH_3111 is not applicable during FOTA session.. ~~Note2: TECH_3111 is not applicable for support check.~~ Note3: If TECH_3111 is applied and this service is used in plant, ECU designers shall apply Virgin mode (Refer to Section 11).						

Threat						
TH_A_3101(1)	In case of other case than TH_I_3101, ex when the vehicle is moving, the illegitimate control of actuators might lead to unexpected action of actuators, which might lead to unsafety.					
Security requirements				Flexibility		
				S_ECU		
FUNC_ID	Functional Requirement	TECH_ID	Technical Requirement	SR	NS	NS_ECU
FUNC_312x(1)	Determine the execution conditions	TECH_3121(1)	The ECU SHALL start RoutineControl service, only when the vehicle is in the safe condition.	CF0	NA	NA
		TECH_3122(1)	The ECU SHALL stop RoutineControl service at the timing when the state of the vehicle is changed to any other conditions than the safe condition.			
FUNC_313x(1)	Control the service access	TECH_3131(1)	The ECU SHALL be able to start RoutineControl service after getting unlock status by SecurityAccess 0x27 (Refer to chapter 10)	CF0	NA	NA
Note1: (TECH_3121 and TECH_3122) or TECH_3131 has/have to be met. Note2: (TECH_3121 and TECH_3122) is recommended if this service/parameter is used in no network environment. ~~Note3: TECH_3131 is not applicable for support check.~~ Note4: If TECH_3131 is applied and this service is used in plant, ECU designers shall apply Virgin mode (Refer to Section 11).						

6.15 RequestDownload (0x34), RequestUpload (0x35), TransferData (0x36), RequestTransferExit (0x37) and RequestFileTransfer (0x38) service

<table>
<tr><th colspan="4">Threat</th><th colspan="2"></th></tr>
<tr><td>TH_C_3400(1)</td><td colspan="5">Out of scope</td></tr>
<tr><td>TH_A_3400(1)</td><td colspan="5">There is no threat for unsafety.</td></tr>
<tr><td colspan="4">Security requirements</td><td colspan="2">Flexibility</td></tr>
<tr><td>FUNC_ID</td><td>Functional Requirement</td><td>TECH_ID</td><td>Technical Requirement</td><td>S_ECU</td><td>NS_ECU</td></tr>
<tr><td colspan="4">No Requirement</td><td></td><td></td></tr>
</table>

<table>
<tr><th colspan="8">Threat</th></tr>
<tr><td>TH_I_3401(1)</td><td colspan="7">Malicious firmware might be written into ECUs.</td></tr>
<tr><td colspan="4">Security requirements</td><td colspan="4">Flexibility</td></tr>
<tr><td rowspan="2">FUNC_ID</td><td rowspan="2">Functional Requirement</td><td rowspan="2">TECH_ID</td><td rowspan="2">Technical Requirement</td><td colspan="2">S_ECU</td><td colspan="2">NS_ECU</td></tr>
<tr><td>F_Se</td><td>DT_Se</td><td>F_Se</td><td>DT_Se</td></tr>
<tr><td>FUNC_344x(1)</td><td>Control the service access</td><td>TECH_3441(1)</td><td>The ECU SHALL be able to start RequestDownload (0x34), RequestUpload (0x35), TransferData (0x36), RequestTransferExit (0x37) and RequestFileTransfer (0x38) services after getting unlock status by SecurityAccess 0x27 (Refer to chapter 10)</td><td>NA</td><td>F0</td><td>NA</td><td>F0</td></tr>
<tr><td colspan="8">Note1: TECH_3441 is not applicable during FOTA session.
Note2: If TECH_3441 is applied and this service is used in plant, ECU designers shall apply Virgin mode (Refer to Section 11).</td></tr>
</table>

6.16 CommunicationControl (0x28), ControlDTCSetting (0x85), LinkControl (0x87), ResponseOnEvent (0x86), ReadScalingDataByIdentifier (0x24), ReadDataByPeriodicIdentifier (0x2A), SecuredDataTransmission (0x84) and AccessTimingParameter (0x83) service

Threat					
TH_2801(1)	These services are forbidden in [REF9]. When the diagnostic tool requests these diagnostic services, ECUs might act unexpectedly, which might lead to unsafety.				
Security requirements				Flexibility	
FUNC_ID	Functional Requirement	TECH_ID	Technical Requirement	S_ECU	NS_ECU
FUNC_280x(1)	Work as designed	TECH_2801(1)	The ECUs SHALL work as the requirements defined in ECU specification and SHALL send a response following [REF1 and REF9]. (e.g. Negative response code = 0x11, 0x7F…)	F0	F0

7. Security requirements for protecting firmware

[FUNC_2901(2)]

All reprogrammable ECUs SHALL implement firmware signature as specified in [Section 6 of REF16]."

8. Security requirements for supplier diagnostic specification

This section describes the security requirements for systemSupplierSpecific Session, other session specified by ECU suppliers, supplier specific diagnostic services and diagnostic service parameters defined by suppliers.

Threat					
TH_D101(1)	Supplier specific diagnostic functions used for pre-production (development) may be present in post-production (aftersales) without sufficient security mitigations. These could be used by an attacker to bypass existing security or safety mechanisms.				
Security requirements				Flexibility	
FUNC_ID	Functional Requirement	TECH_ID	Technical Requirement	S_ECU	NS_ECU
FUNC_D10x(1)	Delete the software for development	TECH_D101(1)	The supplier specific diagnostic services, diagnostic service parameters defined by suppliers and systemSupplierSpecificSession SHALL be deleted for production software.	CF0	CF0
FUNC_D11x(1)	Control the session access	TECH_D111(1)	The supplier specific diagnostic services and diagnostic service parameters defined by suppliers SHALL only be allowed during systemSupplierSpecific Session or other session specified by ECU suppliers.	CF0	CF0
		TECH_D112(1)	systemSupplierSpecific Session and other session specified by ECU suppliers SHALL be started after getting unlock status by OEM SecurityAccess 0x27 service (1^{st} priority, Refer to chapter 10) or supplier specific SecurityAccess 0x27 service (2^{nd} priority).		
		TECH_D113(1)	The contents of the supplier specific SecurityAccess 0x27 service specification, which include, but are not limited to the following, SHALL be reviewed by the security team. · How to protect from dictionary attacks? · How to protect from brute force attacks? · How to protect from replay attacks? · How to protect from predictive analytics cyber-attacks?		
		TECH_D114(2)	Only one SecurityAccess shall be selected in OEM SecurityAccess 0x27 service (1^{st} priority, Refer to chapter 10) or supplier specific SecurityAccess 0x27 service (2^{nd} priority).		
FUNC_D12x(1)	Control the service access	TECH_D121(1)	The supplier specific diagnostic services and diagnostic service parameters defined by suppliers SHALL be started after getting unlock status by OEM SecurityAccess 0x27 service (1st priority, Refer to chapter 10) or supplier specific SecurityAccess 0x27 service (2nd priority).	CF0	CF0

		TECH_D122(1)	The contents of the supplier specific SecurityAccess 0x27 service specification, which include, but are not limited to the following, SHALL be reviewed by the security team. · How to protect from dictionary attacks? · How to protect from brute force attacks? · How to protect from replay attacks? · How to protect from predictive analytics cyber-attacks?		
		TECH_D123(2)	Only one SecurityAccess shall be selected in OEM SecurityAccess 0x27 service (1st priority, Refer to chapter 10) or supplier specific SecurityAccess 0x27 service (2nd priority).		
NOTE1: TECH_D101 or (TECH_D111, TECH_D112 and TECH_D113 and TECH_D114) or (TECH_D121 and TECH_D122 and TECH_D123) has/have to be met. NOTE2: TECH_D101 is recommended strongly. (TECH_D111, TECH_D112 and TECH_D113 and TECH_D114) or (TECH_D121 and TECH_D122 and TECH_D123) are alternative solutions. NOTE3: Supplier specific SecurityAccess's security responsibility is supplier.					

9. Security requirements based on the principle of least privilege

This section describes the security requirements for the diagnostic service with an unnecessary CAN ID and logicalAddress, unnecessary diagnostic services and unnecessary diagnostic services parameters based on the principle of least privilege.

9.1 CANID on DoCAN

Threat					
TH_D200(1)	When the diagnostic tool sends requests with other ECU CANID, ECUs might act unexpectedly, which might lead to unsafety.				
Security requirements				Flexibility	
FUNC_ID	Functional Requirement	TECH_ID	Technical Requirement	S_ECU	NS_ECU
FUNC_D20x(1)	Work as designed	TECH_D201(1)	The ECUs SHALL work as the requirements defined in ECU specification and SHALL not send a response following [REF7].	F0	F0

9.1 LogicalAddress on DoIP

Threat					
TH_D210(1)	When the diagnostic tool sends requests with other ECU logicalAddress, ECUs might act unexpectedly, which might lead to unsafety.				
Security requirements				Flexibility	
FUNC_ID	Functional Requirement	TECH_ID	Technical Requirement	S_ECU	NS_ECU
FUNC_D21x(1)	Work as designed	TECH_D211(1)	The ECUs SHALL work as the requirements defined in ECU specification and SHALL send a NACK code following [REF13]	F0	F0

9.2 Diagnostic services and diagnostic service parameters for Extended Diagnostics

Threat					
TH_D220(1)	When the diagnostic tool requests not supported diagnostic services or with not supported diagnostic services parameters, ECUs might act unexpectedly, which might lead to unsafety.				
Security requirements				Flexibility	
FUNC_ID	Functional Requirement	TECH_ID	Technical Requirement	S_ECU	NS_ECU
FUNC_D22x(1)	Work as designed	TECH_D221(1)	The ECUs SHALL work as the requirements defined in ECU specification and SHALL send a response following [REF1 and REF9]. (e.g. Negative response code = 0x11, 0x12, 0x7E…)	F0	F0

10. Security requirements for 0x27 service (Asymmetric key)

This section describes requirement for SecurityAcess 0x27 service (Asymmetric key). The section 10.3 is 0x27 service using RSA 2048 base. ~~The section 10.4 is 0x27 service using ECDSA base.~~

10.1 Purpose

The ECU judges if diagnostic tool is proper or not by authentication.

0x27 service using Asymmetric key is described in this section to have a stronger method than [REF10].

10.2 Function outline

At first, as general expression, this section explains the way of authentication.

The ECU judges if diagnostic tool connected via DLC or bus which is converted from DoIP(DLC) is proper tool or not, using challenge / response with asymmetrical signature.

Tool and server execute IP communication via internet or intranet.

Authentication is executed using 4 steps. (in Fig.7 —Authentication outline)

- Communication port between server and tool is encrypted by using encryption protocol such as TLS.
- Server authenticates user using “User authentication key 1“(in Fig.8 —Authentication key) and User authentication key inputted by user via tool.
- As next step, Server authenticates tool using “Tool authentication key 1 and 2” (in Fig.8 —Authentication key)
- As next step, The ECU validates server signature using challenge / response mechanism.

In validation process, server uses “Server’s private key” and the ECU uses “Server’s public key” (in in Fig.8 —Authentication key).

The ECU authenticates it is proper server, as these result, the ECU judges it is proper tool used by the proper user.

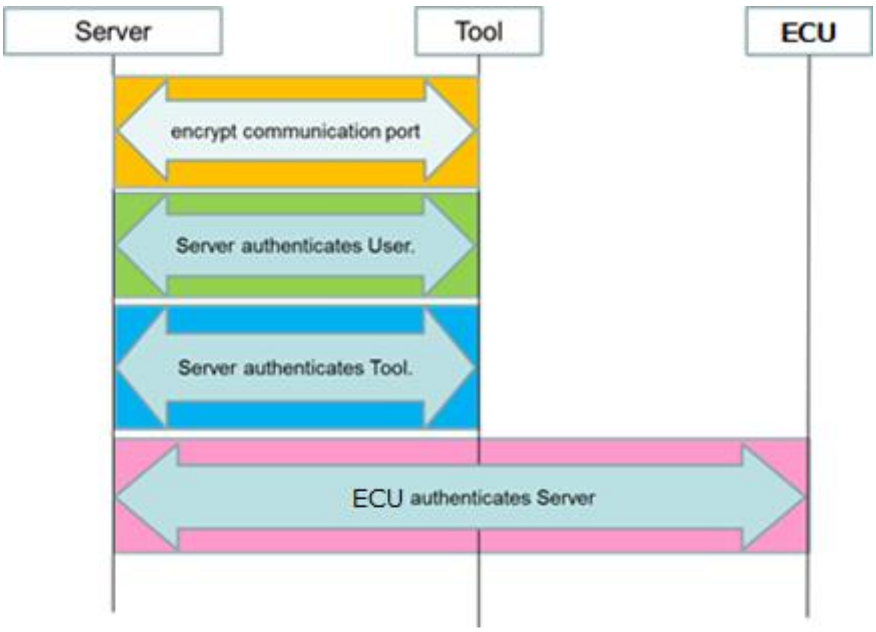


Fig.7 —Authentication outline

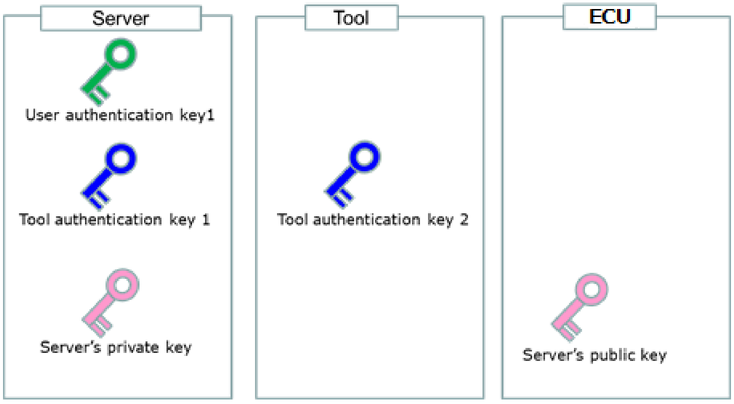


Fig.8 —Authentication key

10.3 Authentication method requirement (RSA 2048 base)

Authentication method : Extended Security Access method

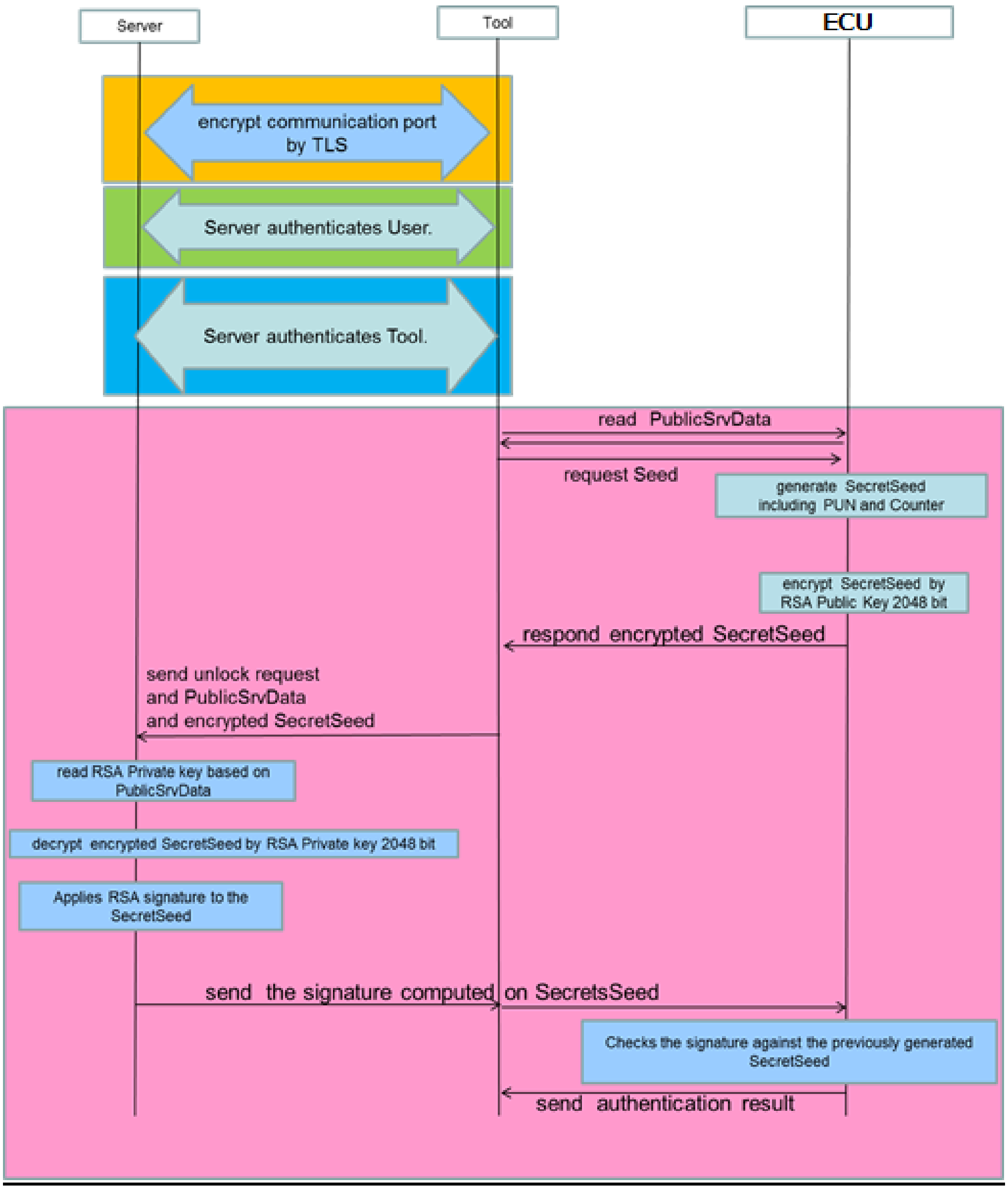


Fig.9 —Authentication sequence of extended Security Access method(RSA 2048)

[FUNC_2701(1)]

SecurityAccess(RSA 2048) shall be done according to [REF10] and this section.

This section are duplicate with [REF10]. (This section is more secured authentication method.)

ECU shall prefer this section to [REF10].

[FUNC_2702(2)]

Security access shall not be done if a memory failure is detected while reading counter/random number (diagnostic service response shall be negative 0x22).

10.3.1 Read PublicSrvData

[FUNC_2703(2)]

The ECU shall send *PublicSrvData* after receiving request to read *PublicSrvData* from Tool.

PublicSrvData is same as *IndexSrvdata.* (DID=F011, refer to [REF17])

10.3.2 Generate SecretSeed

[FUNC_2704(1)]

The ECU shall generate *SecretSeed* after receiving request of Seed from Tool.

[FUNC_2705(1)]

PUN included in *SecretSeed* shall be 256 bits length random number

[FUNC_2706(1)]

Counter included in *SecretSeed* shall be 256 bits length incremental counter.

[FUNC_2707(1)]

SecuredSrvData included in *SecretSeed* shall be unique serial number for each ECU.

[FUNC_2708(1)]

The format of *SecretSeed* (86bytes) shall follow to Table 10-1.

Table 10-1 —SecretSeed Format

#	Data name	Means	Byte position [byte]	Byte length [byte]
#1	PUN	Random Number	0	32
#2	Counter	Incremental counter	32	32
#3	SecLev	Sub-function parameter for “$27 - SecurityAccess”	64	1
#4	SessionID	Session ID parameter for “$10 - StartDiagnosticsSession”	65	1
#5	SecuredSrvData	ECU serial number (DID: F18C)	66	20

SecretSeed byte order is big endian.

(Refer to [REF10] for explanation of parameters)

10.3.3 Encrypt SecretSeed

[FUNC_2709(1)]

The ECU shall have Server’s public key (RSA 2048 bits).

[FUNC_2710(1)]

The ECU shall encrypt *SecretSeed* by Server’s public key.

[FUNC_2711(1)]

Encryption and padding of SecretSeed shall follow RSA-OAEP.

[FUNC_2712(1)]

Detail information of RSA-OAEP is described in later section.

[FUNC_2713(1)]

The ECU shall send encrypted *SecretSeed* to the diagnostic tool as response for request of Seed.

10.3.4 Verify SecurityKey

[FUNC_2714(1)]

The ECU shall verify the SecurityKey sent by the server, which is actually the RSASSA-PSS signature, against the SecretSeed. This signature is 256 bytes (2048 bits). If the signature check is not OK, the ECU shall reject the authentication.

[FUNC_2715(1)]

The ECU shall verify the signature with the server's public key.

[FUNC_2716(1)]

Verification of the signature shall follow RSASSA-PSS.

<u>Example</u> :

With OpenSSL, the signature could be done this way :

openssl dgst -sha256 -sigopt rsa_padding_mode:pss -sigopt rsa_pss_saltlen:0 -sign privKey.key -out signature.sig data.bin

And the verification this way :

openssl dgst -sha256 -sigopt rsa_padding_mode:pss -sigopt rsa_pss_saltlen:0 -signature signature.sig -prverify privKey.key data.bin

- This is only an example and these commands <u>shall no be used as it is.</u>

[FUNC_2717(1)]

Detail information of RSASSA-PSS is described in later section.

[FUNC_2718(1)]

The ECU shall send result of verification.

10.3.5 Key storage

[FUNC_2719(1)]

Server's public key which the ECU has shall be put in area which is not deleted by reset and reprogramming.

[FUNC_2720(2)]

Server's public key shall be protected against change by unauthorized way using software or hardware mechanism (OTP/Secure strage/HSM).

10.3.6 RSA-OAEP

[FUNC_2721(1)]

RSA-OAEP shall follow PKCS#1 v2.2 (see RFC 8017 - 2016).

[FUNC_2722(1)]

Parameters for RSA-OAEP shall follow below (refer to chapter 7.1 RSA-OAEP in PKCS#1).

RSAES-OAEP-ENCRYPT:

- Hash: SHA-256
- MGF: MGF1 (refer to chapter B2.1 in PKCS#1, Hash = SHA-256)
- n = the RSA modulus,
- e = the RSA public exponent,
- M = SecretSeed

- L = "" (empty string)
- C = RSA-OAEP Encrypted SecretSeed (the output)

RSAES-OAEP-DECRYPT (the scope of the diagnostic tool/sever):

- Hash: SHA-256
- MGF: MGF1 (refer to chapter B2.1 in PKCS#1, Hash = SHA-256)
- K = d (the RSA private exponent) and n (the RSA modulus)
- C = RSA-OAEP Encrypted SecretSeed
- L = "" (empty string)
- M = SecretSeed (the output)

10.3.7 RSASSA-PSS

[FUNC_2723(1)]

RSASSA-PSS shall follow PKCS#1 v2.2 (see RFC 8017 - 2016).

[FUNC_2724(1)]

Parameters for RSASSA-PSS shall follow below (refer to chapter 8.1 in RFC 3447).

RSASSA-PSS-SIGN (the scope of the diagnostic tool/sever):

- Hash: SHA-256,
- MGF: MGF1 (refer to chapter B2.1 in PKCS#1, Hash = SHA-256),
- sLen = 32,
- K = d (the RSA private exponent) and n (the RSA modulus),
- M = SecretSeed,
- S = SecurityKey (parameter specific to the signature verification operation).

RSASSA-PSS-VERIFY:

- Hash: SHA-256,
- MGF: MGF1 (refer to chapter B2.1 in PKCS#1, Hash = SHA-256),
- sLen = 32,
- n = the RSA modulus,
- e = the RSA public exponent,
- M = SecretSeed,
- S = SecurityKey (the signature to be checked).

10.3.8 Generating Counter

[FUNC_2725(1)]

The value of the generated Counter shall be stored in a dedicated space in the non volatile memory and shall overwrite the previously stored value.

[FUNC_2726(1)]

Each new value of the Counter must be equal to the previous one + 1.

[FUNC_2727(1)]

Initialization of the Counter shall be 0 at the first time (no previously stored value).

[FUNC_2728(2)]

If the counter reaches its maximum value, authentication shall be done but authentication shall not be successful (diagnostic service response is negative NRC=22 conditionsNotCorrect).

[FUNC_2729(1)]

The counter must be rewritten to the non volatile memory as soon it is incremented (that is to say, for each security access).

10.3.9 Generating PUN (Random number) requirement

[FUNC_2730(2)]

The ECU generates CSPRNG (cryptographically secure pseudo random number generator). This requirement follows NIST recommendation: SP800-90Ar1. (It's not necessary to refer SP800-90B/C)

10.3.10 Anti-brute force

[FUNC_2731(2)]

The ECU shall implement a flag to set the status (success/fail) of the authentication in non-volatile memory as soon as an authentication validation fails (content of securityKey is not correct).

The ECU shall store the number of consecutive failed authentication attempts (Att_Cnt) in non-volatile memory and increment it as soon as an authentication validation fails.

This counter shall be reset to 0 as soon as a successful authentication happens.

This counter shall be reset to 0 as soon as a delay timer expiration occurs.

After a reboot from the ECU, the value of Att_Cnt shall be retrieved from NVM.

After 10 consecutive failed validations (Att_Cnt >= Att_Cnt_Limit; where Att_Cnt_Limit=10), the ECU shall set a delay before answering positively to the next requestSeed.

This delay should be set at 16 minutes. (Delay_Timer=16 minutes).

Timer shall start after the 10th consecutive validation failure.

Delay_timer shall continue through diagnostic session changes.

At ECU power on/start up, if Att_Cnt>Att_Cnt_Limit, the delay timer shall be invoked.

During the delay period the ECU must respond negatively to Seed requests with NRC 0x37.

[FUNC_2732]

SecretSeed (including PUN, counter, …etc) shall be same value before set the status (success/fail) of the authentication validation. In next authentication after set the status (success/fail) of the authentication validation, SecretSeed shall be new value. (Static_Seed=true)

10.3.11 key for pre-production phases (development) and post-production phases (aftersales)

[FUNC_2733(1)]

ECU shall manage different keys and different PublicSrvData for pre-production phases (development) and post-production phases (aftersales).

[FUNC_2734(1)]

As keys for post-production phases, the ECU shall manage different keys by each kind of ECU x each supplier.

[FUNC_2735(1)]

For pre-production phases (development), ECU shall use the same public key and PublicSrvData value (PROTASYMV2) as the C1AHS Gateway.

11.F-VIRGIN mode and VIRGIN mode

11.1 F-VIRGIN mode and VIRGIN mode

The F-VIRGIN mode is for pre-production phases (development). When the ECU is in the F-VIRGIN mode, the ECU uses dummy SecurityAccess described in Section 11.1.6. In F-VIRGIN mode, a diagnostic tool can access secured services/ functions/ data without having to calculate SecurityKey nor using a decryption algorithm on SecuritySeed. ECU can go back to F-NORMAL mode using reprograming for calibration. This calibration shall be managed secretly in ECU designer/supplier.

The VIRGIN mode is for production phases (plant). When the server is in the VIRGIN mode, the ECU uses dummy SecurityAccess described in Section 11.1.6. So, diagnostic tool can accesssecured services/functions/data without having to calculate SecurityKey nor using a decryption algorithm on SecuritySeed. ECU can go back to NORMAL mode in two ways:

1. A diagnostic tool writes the NORMAL value in the {Virgin Flag} DID using the $2E UDS service
2. If DitanceTotalizeris strictly higher than 500km, the ECU shall automatically switch to NORMAL mode.

- VIRGIN mode implementation is mandatory, if production phases (plant) use diagnostic function protected by SecurityAccess (Asymmetric key).
- F-VIRGIN mode implementation is mandatory if ECU designer/supplier hope to use in pre-production phases (development)

11.1.1 Description

[FUNC_2801(2)]

The description of labels used in this section is written in the following table.

Table 11-1 — Label description

Label		Meaning		
Calibration/firmware information	{F-Virgin flag}	0x00	F-VIRGIN	When the ECU is in the F-VIRGIN mode, the ECU SHALL ignore the value of {Virgin Flag} and always allow dummy SecurityAccess described in Section 11.1.6. As a result the secured services/ functions/data can be accessed without calculation of SecurityKey nor usage of encrypted algorithm by the tool in the F-VIRGIN mode.
		0xFF	F-NORMAL	F-NORMAL is the default value of {F-Virgin flag}, which means value of {Virgin flag} is effective.
nonvolatile memory information	{Virgin flag}	0x00	VIRGIN	VIRGIN is the default value of {Virgin flag}, which means the ECU is in the VIRGIN mode. When the ECU is in the VIRGIN mode, the ECU SHALL allow dummy SecurityAccess described in Section 11.1.6. As a result that secured services/functions/data can be accessed without calculation of SecurityKey and usage of encrypted algorism in the VIRGIN mode.
		0xFF	NORMAL	When the ECU is in the NORMAL mode, the ECU

				SHALL NOT use dummy SecurityAccess referring to Section 11.1.6. As a result that secured services functions/data can be accessed after getting unlocked through formal SecurityAccess described in Section 10.
nonvolatile memory information	{Mileage_for_NormalMode} boolean	True		{Mileage_for_NormalMode} shall be computed every 1 second while {Mileage_for_NormalMode} is false {Mileage_for_NormalMode} is set to true and immediately stored in NVM when DistanceTotalizer exceeds 500km and is different from invalid value
		False		Default Value
CAN/ETHERNET information	DistanceTotalizer	Current Distance traveled by the vehicle		

11.1.2 {F-Virgin flag} Setting using reprograming command

[FUNC_2802(2)]

When the ECU is reflashed with calibration or firmware including {F-Virgin flag}, the ECU SHALL follow these requirements. First case is behavior in F-NORMAL mode, Second case is behavior in F-VIRGIN mode.

- If {F-Virgin flag} is F-NORMAL then Value of {Virgin flag} is effective.
  - So, if {Virgin flag} is NORMAL, the ECU SHALL use formal SecurityAccess referring to Section 10.
  - So, if {Virgin flag} is VIRGIN, the ECU SHALL use dummy SecurityAccess referring to Section 11.1.6.
- If {F-Virgin flag} is F-VIRGIN, then the ECU SHALL ignore the value of {Virgin_Flag}..
  - So, if {F-Virgin flag} is F-VIRGIN, the ECU SHALL use dummy SecurityAccess referring to Section 11.1.6.

11.1.3 {Virgin flag} Setting using virgin flag command and Mileage

[FUNC_2809(1)]

The ECU SHALL implement the DID F062 to store the value of {Virgin Flag} according to Table 11-2

Table 11-2 — Data Identifiers for {Virgin Flag}

Variable	DID	Authorizations	Values
{Virgin Flag}	0xF062	● {Mileage_for_NormalMode} =false ➔ Read/Write ● {Mileage_for_NormalMode} =true ➔ Read only	0x00: VIRGIN 0xFF: NORMAL Other: forbidden

[FUNC_2810(1)]

The value of {Virgin Flag} DID SHALL be initialized to VIRGIN.

[FUNC_2811(1)]

If {Mileage_for_NormalMode} = true

- The ECU SHALL force the value of {Virgin Flag} DID to NORMAL.
  - If {F-Virgin Flag} = F-NORMAL: ECU SHALL always use Formal SecurityAccess described in

Section 10
    - If {F-Virgin Flag} = F-VIRGIN: ECU SHALL always use dummy SecurityAccess described in Section 11.1.6

[FUNC_2812(1)]

If {Mileage_for_NormalMode} = false

- The ECU SHALL answer positively to READ {Virgin Flag} DID in default session of applicative software.
- The ECU SHALL answer positively to READ {Virgin Flag} DID in extended session of applicative software.
- The ECU SHALL answer positively to READ {Virgin Flag} DID in all sessions of bootloader software.
- The ECU SHALL answer positively to WRITE {Virgin Flag} DID in all sessions of bootloader software and in extended session of applicative software.
    - In accordance with section 6.9.1, as the service WriteDataByIdentifier ($2E) is used to change the value of {Virgin Flag}, this command SHALL only be accepted when ECU is in unlocked status. In the case of {Virgin Flag}, this means:
        - If {F-Virgin Flag} = F-NORMAL
            - If {Virgin flag} is <u>VIRGIN</u>, the ECU SHALL use dummy SecurityAccess described in Section 11.1.6.
            - if {Virgin flag} is <u>NORMAL,</u> the ECU SHALL use formal SecurityAccess described in Section 10.
        - If {F-Virgin Flag} = F-VIRGIN
            - The ECU SHALL use dummy SecurityAccess described in Section 11.1.6.

[FUNC_2813(1)]

If {Mileage_for_NormalMode} = true

- The ECU SHALL answer positively to READ {Virgin Flag} DID in default session of applicative software.
- The ECU SHALL answer positively to READ {Virgin Flag} DID in extended session of applicative software.
- The ECU SHALL answer positively to READ {Virgin Flag} DID in all sessions of bootloader software.
- The ECU SHALL answer negatively with NRC 0x22 to WRITE {Virgin Flag} DID in all sessions of bootloader and applicative software.

Examples for request and response message of virgin flag writing command are defined as following.

Table 11-3 defines the request message of virgin flag command.

Table 11-3 — Request message of virgin flag writing command definition

Parameter Name	Byte value
WriteDataByIdentifier Request SID	0x2E
dataIdentifier[] = [byte#1 byte#2]	 0xF0 0x62
dataRecord[] = [date#1]	 0x00 : VIRGIN 0xFF : NORMAL

	Other : forbidden

Table 11-4 defines the positive response message of virgin flag writing command.

Table 11-4 — Positive response message of virgin flag command definition

Parameter Name	Byte value
WriteDataByIdentifier Response SID	0x6E
dataIdentifier[] = [	
byte#1	0xF0
byte#2]	0x62

Table 11-5 defines the negative response message of virgin flag writing command.

Table 11-5 — Negative response message of virgin flag command definition

Parameter Name	Byte value
Negative Response SID	0x7F
WriteDataByIdentifier Request SID	0x2E
responseCode	0xXX

11.1.4 Performance requirement for virgin flag command

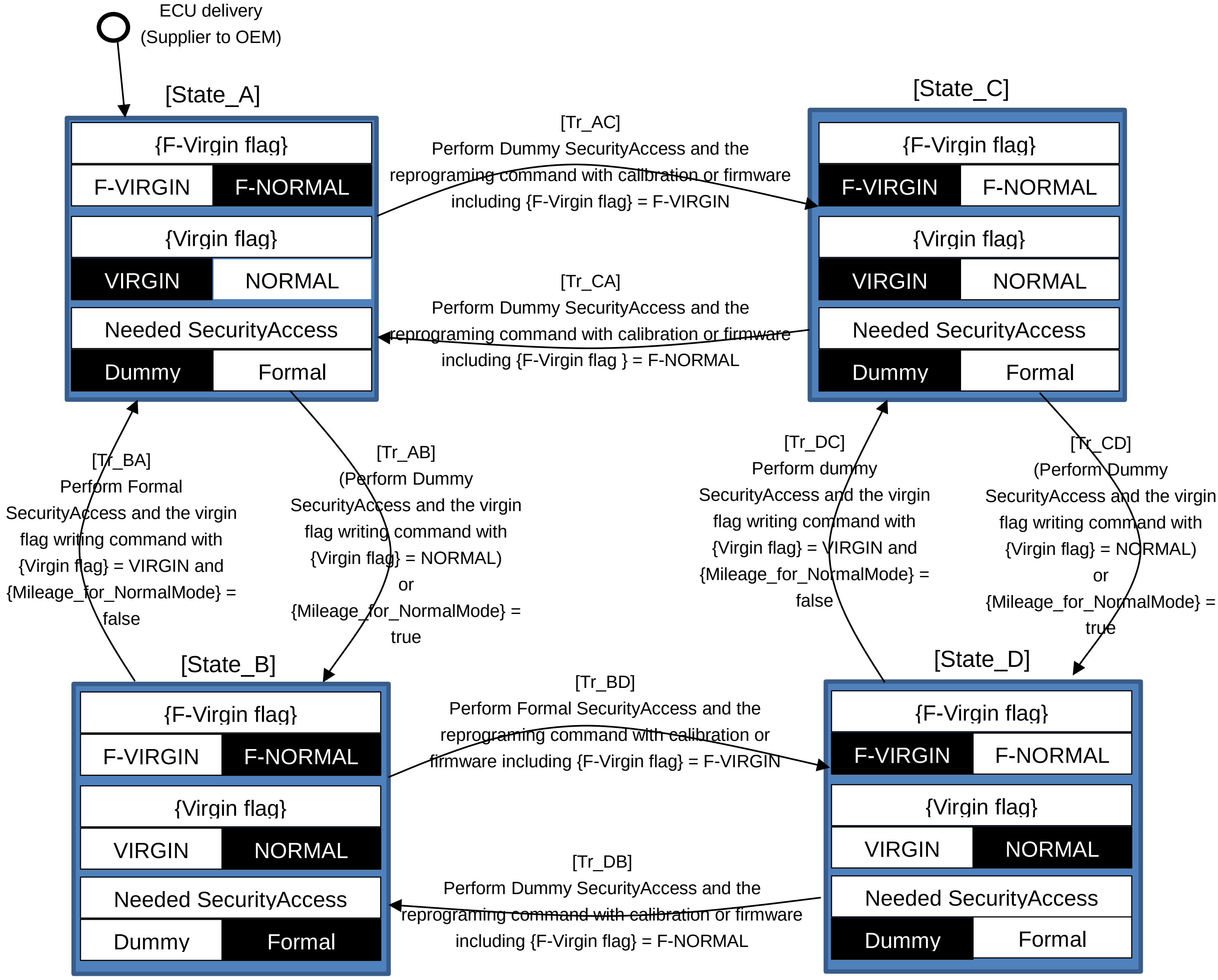
[FUNC_2804(2)]

When ECU receives writing request (2E/F062/NORMAL) for virgin flag command, ECU shall answers positively if conditions are meet and ensure {Virgin flag} is written in NVM. Performance (P2 time) for virgin flag command shall be within 100ms.

11.1.5 State transition diagram

[FUNC_2805(2)]

The following is State transition diagram for F-VIRGIN/F-NORMAL/VIRGIN/NORMAL mode and lock/unlock status.



11.1.6 Dummy SecurityAccess in case of VIRGIN mode or F-VIRGIN mode

[FUNC_2806(2)]

In case of VIRGIN mode or F-VIRGIN mode,

When ECU receive SecurityAccess request for requestSeed, ECU always shall send positive response using the following securitySeed

In case that SecurityAccess status is “unlock” : securitySeed that all value are 0x00 and length 256bytes.

In case that SecurityAccess status is “lock” : securitySeed that all value are 0xFF and length 256bytes.

When ECU receive SecurityAccess request for sendKey with any securityKey, ECU always shall send positive response. SecurityAccess status is changed from lock to unclock.

Since any securityKey shall be accepted by the ECU during a dummy Security Access, Att_Cnt value shall never be increased during a dummy Security Access.

This request can be realized less impact to remove calculation of SecurityKey and usage of encrypted algorism from tool (ex. plant equipment)

For Dummy SecurityAccess, each individual request shall be handled within 50ms by ECU.

In case of not VIRGIN mode and not F-VIRGIN mode, please follow to section 10.

	Request	Expected Response	
SecurityAccess request for requestSeed	27 01	SecurityAccess status is “unlock”	67 01 00 00 00 00 00 00 00 00 …………(all value for securitySeed are 0x00 and length is 256bytes)
		SecurityAccess status is “lock”	67 01 FF FF FF FF FF FF FF FF …………(all value for securitySeed are 0xFF and length is 256bytes)
SecurityAccess request for sendKey	27 02 XX XX …… (with any securityKey)	67 02	

11.1.7 Request of dummy SecurityAccess and virgin flag command execution

[FUNC_2808(2)]

Based on cybersecurity strategy regarding regulations, in order to exit the VIRGIN mode, the plants SHOULD be required to execute dummy SecurityAccess and virgin flag command before the post-production phase. .